УДК 004.383

СЕРГІЄНКО А. М.,
КЛИМЕНКО І. А.,
СЕРГІЄНКО П. А.

РЕКОНФІГУРОВАНА БАГАТОПРОЦЕСОРНА ОБЧИСЛЮВАЛЬНА СИСТЕМА НА ПЛІС

Запропоновано підхід до проектування реконфігурованих обчислювальних систем на програмованих логічних інтегральних схемах, який ґрунтується на відображенні графів синхронних потоків даних у багатопроцесорну систему, в якій реконфігурація виконується через перемикання потоків даних. Для реалізації процесорних елементів такої системи розроблено шістнадцятирозрядне ядро RISC-процесора, який має малі апаратні витрати та систему команд, що настроюється.

An approach to the reconfigurable computer design based on FPGAs is proposed. It is based on mapping the synchronous dataflow graphs into the many-core processor system in which the reconfiguration means the dataflow switshing. A soft core of the 16-bit RISC microprocessor is designed for this system, which has small hardware volume and configurable instruction set.

Ключові слова: ПЛІС, конфігурований комп′ютер, RISC.

1. Вступ

Впровадження програмованих логічних інтегральних схем (ПЛІС), як основи конфігурованих комп′ютерів, є сталою тенденцією. ПЛІС, в якій реалізовано високопродуктивний спеціалізований обчислювач, програмується один раз для виконання певної задачі або вона повністю чи частково перепрограмується кілька разів під час виконання цієї задачі, забезпечуючи оптимальну обчислювальну структуру для різних фрагментів алгоритму. В останньому випадку кажуть про динамічно реконфігурований комп′ютер (ДРК) [1,2,3].

Як правило, в ДРК реалізуються потокові алгоритми, для яких можна передбачити об′єми порцій обчислень та міжпроцесорні комунікації. При розробці ДРК алгоритм моделюють, виділяють множини обчислень, які відображають на процесорні елементи (ПЕ), визначають потрібні варіанти процесорних структур та виробляють набір конфігураційних файлів, які завантажуються динамічно у ПЛІС під час роботи ДРК [1].

Нові серії ПЛІС фірм Altera і Xilinx підтримують часткову реконфігурацію. Але, по-перше, мінімальна порція реконфігурації покриває прямокутну область ПЛІС, яка є доволі великою. Наприклад, в ПЛІС Xilinx ця область включає чотири блоки множення DSP48. Крім того, частина ПЛІС, яка не змінила конфігурацію і острівець з новою конфігурацією повинні мати незмінні спільні точки передачі даних. Все це призводить до складнощів проектування ДРК та обмежень до оптимізації апаратних витрат. По-друге, мінімальна часткова реконфігурація триває кілька мілісекунд, тобто, вона складає великі часові накладні витрати [2].

Ці накладні витрати можна мінімізувати завдяки кешуванню прошивки [3]. Але у ПЛІС на зберігання прошивки вже використовується близько 60% транзисторів кристалу. Тому додаткові апаратні витрати на збереження прошивки призводять як до нераціонального використання ресурсів ПЛІС, так і до зменшення швидкодії конфігурованих схем [2].

Нові серії ПЛІС мають ряд особливостей, що впливають на вибір архітектури модулів, які в них конфігуруються. Якщо число транзисторів у ПЛІС, збільшується учетверо, то об′єм ресурсів для трасування сигналів збільшується лише удвічі. Затримки у лініях зв′язку стали в 1–3 рази більшими за затримки у логічних таблицях (ЛТ) та інших логічних елементах [2]. Як результат, конфігуровані модулі мають бути компактними, щоби їх внутрішні сигнали далеко не поширювались.

Часові та апаратні накладні витрати на реконфігурацію ДРК можна скоротити, якщо при цьому взагалі не змінювати прошивку ПЛІС. У багатьох роботах, наприклад в [4], пропонується багатопроцесорна архітектура ДРК, яка складається з множини однакових компактних ПЕ, які зв′язані конфігурованою системою комутації. Така система може швидко перепрограмуватись динамічно. Недоліком цієї архітектури є той, що ПЕ є доволі спеціалізованими,

що накладає обмеження на множину алгоритмів, що реалізуються.

Як конфігурований ПЕ варто використовувати програмно кероване процесорне ядро зі спеціалізованою системою команд. При реалізації в ПЛІС такий процесор може динамічно змінювати виконуваний алгоритм, завдяки оверлейній структурі програм. Крім того, шляхом часткової реконфігурації можна динамічно змінювати систему команд, не зачіпаючи спільні точки передачі даних.

Вимозі компактності не відповідають ядра мікропроцесорів з поширеною архітектурою, такі як Nios, Microblaze, Mico32, OpenRISC, MIPSFPGA. Для досягнення прийнятної тактової частоти ці процесори мають багатоступеневий конвеєр (5 і більше ступенів). Для виконання кількох незалежних програмних потоків у такому ядрі реалізуються система переривання та зміни контексту, віртуальна пам'ять, кеш-ОЗП, швидкісний інтерфейс до зовнішньої динамічної пам'яті. Для реалізації мультипроцесорної архітектури необхідні комутатори з високою пропускною здатністю та система підтримки режиму когерентності доступу пам'яті [5].

Метою роботи є побудова багатопроцесорного ДРК у ПЛІС, який реалізує модель синхронних потоків даних, завдяки чому зменшуються апаратні витрати, збільшуються швидкодія системи та можливості динамічного реконфігурування. Для цього було розроблене компактне ядро RISC-процесора, яке грає роль ПЕ такої системи.

2. Система синхронних потоків даних

Граф синхронних потоків даних (ГСПД) є зручною моделлю для завдання багатьох алгоритмів з циклічною природою, таких як алгоритми цифрової обробки сигналів. ГСПД складається з вершин, які виконують обчислювальні функції або актори і які з'єднані між собою дугами у систему. Кожен потік-дуга може мати буферну пам'ять типу FIFO. Актор спрацьовує негайно, як тільки є дані на його входах і видає результати на свої виходи. На відміну від графу потоків даних, в ГСПД протягом циклу виконання алгоритму кожен актор споживає і видає на свої виходи одну і ту саму кількість даних.

Модель ГСПД має такі властивості, як відсутність блокувань, статичне складання розкладу, можливість виконання великої множини алгоритмів, простота побудови алгоритмів для неї. Завдяки цьому, вона використовується як для програмування багатопроцесорних систем [6], так і для проектування структур конвеєрних обчислювачів [7, 8, 9].

При одиничному відображенні алгоритму, який задано на ГСПД, структура системи є ізоморфною цьому графу. Кожен процесорний елемент (ПЕ) такої структури виконує єдиний процес обчислення відповідної функції-актора. При появі чергової групи даних у вхідних портах процес запускається, обчислює функцію, записує результати у вихідні порти і зупиняється.

При відображенні ГСПД з N акторів у систему з n ПЕ кожен з них виконує послідовно до $k =]N/n[$ процесів обчислення суміжних акторів. При цьому програмування ПЕ виконується за методами, які обґрунтовані в [6, 10]. Отже, ПЕ системи виконують лише по одному програмному потоку. Для них не потрібна операційна система, система переривань, віртуальна пам'ять, тощо. Тобто, такі ПЕ можуть мати невеликі апаратні витрати і вони будуть компактними.

Цикл обчислення алгоритму ГСПД у такій паралельній системі визначається критичним шляхом передачі і обчислення даних, який проходить через максимально завантажені ПЕ. Решта ПЕ, через які не проходить критичний шлях, будуть недовантаженими [6, 7]. Тому є задача балансування завантаженності ПЕ.

3. Динамічне реконфігурування системи

Задача, яка потребує динамічного реконфігурування, має бути представлена як набір різних ГСПД. Тоді ці ГСПД відображаються у кілька багатопроцесорних структур, які відрізняються кількістю ПЕ та графами їх з'єднань. Ці структури об'єднуються в одну загальну результуючу структуру ДРК, використовуючи відповідні комутаційні вузли.

Завдання, які виконуються у вузлах структури, описуються як програми на відповідній мові програмування і транслюються у машинні коди ПЕ. При цьому виділяється множина команд використаних і віртуальні модулі ПЕ налаштовуються під ці множини, як це запропоновано у роботі [10].

Одержаний проект багатопроцесорної системи компілюється у прошивку ПЛІС разом з програмними кодами за звичайною методикою, використовуючи САПР виробника ПЛІС.

При вирішенні задачі ПЕ у ДРК виконують свої програми, а його реконфігурація полягає лише у відповідній комутації комутаційних

вузлів та у перезавантаженні програм у ПЕ з зовнішньої пам'яті. Такий процес реконфігурації відбувається значно швидше, ніж при повній чи частковій зміні конфігурації ПЛІС.

4. Архітектура ПЕ багатопроцесорної системи

Як ПЕ багатопроцесорної системи синхронних потоків даних, було розроблене 16-розрядне процесорне ядро RISC-ST2. За основу було взяте ядро RISC-ST, яке описане у [9]. Довжина команди була збільшена до 18 розрядів, що дало змогу оптимізувати систему команд та краще задіяти об'єм вбудованої пам'яті ПЛІС.

До ядра додано множину спеціалізованих команд обробки окремих бітів слів, виділення бітових полів заданої довжини, злиття полів, зсуву слів, підрахунку числа нульових старших розрядів. Для реалізації швидкого доступу до асоціативної таблиці введено команду обчислення хеш-функції. Такі команди сприяють ефективній реалізації алгоритму компресії, такому як LZW.

Множина використаних команд програмується під час формування конфігураційного файлу, так як це запропоновано в роботі [10]. Як результат, одержуються невеликий ПЕ і програмні коди скороченого об'єму, які повністю вміщуються всередині ПЛІС.

Регістрова пам'ять процесора має 32 регістри, але командно доступними є 16 молодших регістрів. Щоби максимально задіяти можливості регістрової пам'яті, яка реалізована на ЛТ, у процесорі протягом одного такту виконується читання з трьох регістрів та запис у один регістр.

Використовуються такі види адресації, як регістрова, непряма регістрова, базова, індексна з преінкрементом. Пам'ять даних розбита на сторінки по 256 байтів і має максимальну ємність 16 мегабайт. Для доступу до пам'яті даних використовується стандартний відкритий інтерфейс Wishbone.

Адресно доступними є до 256 периферійних регістрів. Такі регістри слугують регістрами вводу-виводу спецпроцесорів, які виконують швидкісні обчислення, як наприклад, обчислення елементарних функцій, цифрова обробка сигналів, шифрування. Кілька ядер можуть бути об'єднані у систему через їхні регістри вводу-виводу та систему переривань.

Завдяки тому, що конвеєр команд є трьохступеневим, більшість команд виконується за один такт, а команди переходу та читання пам'яті – за два такти. Виклик підпрограм виконується також за два такти, під час чого адреса повернення та прапорці умов зберігаються у апаратному стеку.

Ядро мікропроцесора описане мовою VHDL і не має обмежень для синтезу та конфігурування у ПЛІС будь-якої серії. Модель процесора має вбудований дизасемблер, який спрощує тестування та відлагодження програм. Розроблено програму кросасемблера на мові Java, виходом якої є VHDL-файли блоків пам'яті даних та програм процесора.

5. Реалізація процесорного ядра у ПЛІС

У таблиці 1 показані результати синтезу процесора для ПЛІС Xilinx Kintex-7 та його найближчих поширених аналогів. У порівнянні з аналогами розроблене ядро має менші апаратні витрати і більшу швидкодію. Слід відмітити, що на затримку у лініях міжз'єднання у даному процесорі припадає 75% затримки критичного шляху. Компактність ядра дає змогу розмістити близько трьохсот ядер у ПЛІС середнього об'єму Xilinx xc7k480t, кожне з яких матиме власну пам'ять у 12 кілобайт

Табл. 1. Реалізація мікропроцесорних ядер у ПЛІС Xilinx Kintex-7

Ядро мікропроцесора	Розрядність	Апаратні витрати, ЛТ	Макс. тактова частота, МГц
RISC-ST2	16	653	217
OpenMSP430	16	1387	150
OpenRISC1200	32	4945	107

Ядро RISC-ST2 може бути зконфігуроване у ПЛІС різних серій. В таблиці 2 показані результати такого конфігурування.

Табл. 2. Реалізація ядра RISC-ST2 у ПЛІС

Серія ПЛІС	Апаратні витрати, ЛТ	Макс. тактова частота, МГц
Xilinx Zynq	634	206
Xilinx Spartan6	701	140
Xilinx Artix7	625	150
Altera StratixV	723	240
Altera Cyclone V	999	132
Altera Max10	1522	80
Lattice ECP5U	742	124

Висновки

Запропоновано підхід до створення багатопроцесорних систем на кристалі на основі відображення ГСПД, який дає змогу отримати програмовану високопродуктивну систему для об-

робки потоків даних зі зменшеними апаратними витратами.

Розроблено ядро RISC-процесора зі спеціалізованою системою команд. Завдяки тому, що процесор орієнтований на виконання лише одного обчислювального процесу та обробки переривань, він має невеликі апаратні витрати та високу швидкодію.

На основі цього ядра можлива розробка багатопроцесорних систем на кристалі для обробки потоків даних з продуктивністю у кілька десятків мільярдів операцій за секунду. Планується створення системи, яка виконуватиме швидку декомпресію файлів за алгоритмом LZW.

Список посилань

1. DeHon A. Reconfigurable Computing Architectures / R. Tessier, K. Pocek, A. DeHon // Proceedings of the IEEE. – V. 103. –No. 3. – 2015. – P. 332 –354.
2. Koch D. Partial Reconfiguration on FPGAs. Architectures, Tools and Applications / D. Koch // – New York: Springer. – 2013. – 295 P.
3. Клименко І. А. Оптимізація реконфігурації в динамічно реконфігурованих обчислювальних системах // Вісник НТУУ «КПІ», сер. Інформатика, управління та обчислювальна техніка. – 2015. – №63. – С.93-100.
4. Lanuzza M. MORA: A New Coarse-Grain Reconfigurable Array for High Throughput Multimedia Processing // M. Lanuzza, S. Perri, P. Corsonello // Proc. 7-th Int. Conf. on Embedded Computer Systems, SAMOS'07. – 2007. – P. 159 – 168.
5. Processor Design. System-on-Chip Computing for ASICs and FPGAs / Edited by J. Nurmi. – Springer. – 2007. – 525 p.
6. Lee E. A. Software Synthesis from Dataflow Graphs / S.S. Bhattacharyya, P.K. Murthy, E.A. Lee // – Springer. – 1996. – 190 p.
7. Bhattacharyya S. S. Mapping Parameterized Cyclo-Static Dataflow Graphs onto Configurable Hardware / H. Kee, C.-C. Shen, S. Bhattacharyya, I. Wong, Y. Rao, and J. Kornerup // Journal of Signal Processing Systems. – V. 66. – No 3. – 2012. – P. 285–301.
8. Сергиенко А.М. Алгоритмические модели обработки потоков данных / А.М. Сергиенко, В.П. Симоненко // Электронное моделирование. –2008. –Т.30, №6. –С. 49–60.
9. Сергиенко А.М. VHDL для проектирования вычислительных устройств. – К.: ДиаСофт. – 2003. – 210 с.
10. Sergyienko A. Configurable Microprocessor Array for DSP Applications / O. Maslennikov, Ju. Shevtshenko, A. Sergyienko // LNCS. – Berlin: Springer. – V. 3019. – 2004. – P. 36–41.